



(19) Europäisches Patentamt
European Patent Office
Office européen des brevets



(11) EP 0 528 695 B1

(12)

EUROPEAN PATENT SPECIFICATION

(45) Date of publication and mention
of the grant of the patent:
12.07.2000 Bulletin 2000/28

(51) Int. Cl.⁷: G06F 9/30

(21) Application number: 92307632.7

(22) Date of filing: 20.08.1992

(54) Data processing apparatus

Datenverarbeitungsvorrichtung

Appareil de traitement de données

(84) Designated Contracting States:
DE FR GB NL

• Deguchi, Masashi
Nara-shi, Nara 631 (DE)

(30) Priority: 21.08.1991 JP 20911291

(74) Representative:
Keltie, David Arthur
DAVID KELTIE ASSOCIATES,
12 New Fetter Lane
London EC4A 1AP (GB)

(43) Date of publication of application:
24.02.1993 Bulletin 1993/08

(56) References cited:

(60) Divisional application:
99124499.7 / 0 989 485
99124426.0 / 0 984 358

- IBM TECHNICAL DISCLOSURE BULLETIN. vol. 24, no. 8 , January 1982 , NEW YORK US pages 4018 - 4022 BAKER AND KIRBY 'Instruction execution conditioned on operand addresses'
- IBM TECHNICAL DISCLOSURE BULLETIN. vol. 13, no. 5 , October 1970 , NEW YORK US page 1257 RUBIN 'Add halfword immediate and test instruction'

(73) Proprietor:
MATSUSHITA ELECTRIC INDUSTRIAL CO., LTD.
Kadoma-shi, Osaka-fu, 571 (JP)

(72) Inventors:
• Matsuzaki, Toshimichi
Mino-shi, Osaka 562 (DE)

Note: Within nine months from the publication of the mention of the grant of the European patent, any person may give notice to the European Patent Office of opposition to the European patent granted. Notice of opposition shall be filed in a written reasoned statement. It shall not be deemed to have been filed until the opposition fee has been paid. (Art. 99(1) European Patent Convention).

Description**BACKGROUND OF THE INVENTION****(1) Field of the Invention**

[0001] This invention relates to a data processing apparatus, which is referred to as a central processing unit (CPU), a microprocessor or the like.

(2) Description of the Related Art

[0002] Some conventional data processing apparatuses are constructed so that memory is exclusively used as the operand of operations by operational instructions. In such apparatuses, predetermined operations are executed on the basis of an instruction consisting of an operation code indicating the type of the operation, two operand addresses indicating the locations of data in the memory, for example, a source operand address and a destination operand address.

[0003] Some conventional data processing apparatuses are provided with a register called an accumulator and other registers also, the accumulator being implicitly used as an operand. In such apparatuses, operations are executed on the basis of an instruction consisting of a single register designation code indicating the register to operate besides the accumulator.

[0004] Other conventional data processing apparatuses are provided with general-purpose registers, any of which can be combined to be the operands and operations are executed on the basis of an instruction consisting of an operation code indicating the type of an operation and two register designation codes indicating the register to operate. However, according to the firstly-mentioned apparatuses using memory exclusively as the operand, the instruction word is rather long because it demands an operand field having bit-length to accommodate two operand addresses. As a result, programs written by the use of such apparatuses tend to be large-sized.

[0005] According to the secondly-mentioned apparatuses provided with an accumulator, the instruction word can be short because the instruction merely needs a register field which can accommodate a single register designation code whose bit-length is shorter than that of the operand address. However, operations are assigned mainly to the accumulator, so that frequent moves of data are required between the accumulator and either the other registers or the memory. Such an increase in the number of instructions to be written in programs may decrease the speed of process of the programs as well as increase the program sizes.

[0006] According to the last-mentioned apparatuses provided with general-purpose registers, the number of instructions to be written in programs can be fairly decreased because any combination of the registers can be the operands. The instruction word can be

comparatively short because the bit-length of the register designation code to be used is shorter than that of an operand address.

[0007] Nevertheless, it is difficult for such apparatuses to be constructed so that the sizes of programs can be minimized both by various kinds of operations being executed and they are executed in short instruction word length because of the following problem:

[0008] When a data processing apparatus is provided with, for example, 8 general-purpose registers, the bit-length of the register designation code to specify a register is made 3 bits; the instruction requires a total of 6-bit register field. Thus, 8-bit instruction word length is not practical because usable instructions in that case are at most 4 types while 16-bit instruction word length is actually demanded to get sufficient numbers of instruction types. For example, "Intel i486 MICROPROCESSOR" published in April 1989 by Intel discloses tables showing the instruction format of the Intel i486TM microprocessor. According to this publication, an instruction code format for a register-register move instruction is represented as "1000100W_11AAABBB", and an instruction code format for an immediate-to-register move instruction is represented as "1100011W_11000BBB", where W is a bit indicating whether the data size is represented by bytes, "AAA" indicates a source register, and "BBB" a destination register. It is apparent that these two instructions have different operation codes. This document shows that a specific operation is executed by a single operation code regardless whether or not the two register designation codes (in this case "AAA" and "BBB") are equal.

SUMMARY OF THE INVENTION

[0009] In view of the above problem, an object of this invention is to provide a data processing apparatus, which can minimize the sizes of programs by various types of operations being executed in a short instruction word length.

[0010] The above object can be achieved by a data processing apparatus comprising: decode means for decoding an instruction code including an operation code and a first and a second register designation code; and instruction execution means for executing an appropriate operation according to the results of the decoding by the decode means, wherein the instruction execution means executes a first operation based on the operation code when the second register designation code is different from the first register designation code, and executes a second operation based on the operation code when the second register designation code is equal to the first register designation code, wherein the first operation is executed with first and second registers designated by the first and second register designation codes, the second operation is executed with the first register designated by the first register designation code and immediate data included in the instruction

code, and the first and second operations correspond to the same operation with different addressing modes.

[0011] The instruction execution device may comprise an inquiry device for inquiring as to whether the two register designation codes are different from each other or equal and a control device for controlling the first and second processes so that they can be selective according to the results inquired by the inquiry device.

[0012] If the first process were assigned to a register data move process in which a source register and a destination register are identical, the value of the register would be the same as the value before the execution, however, this type of process is hardly required. Therefore in such a case, two different processes can be assigned to a single operation code by allowing the instruction code to execute the second process.

[0013] Another example of the first process is that if it were assigned to an operation like an adding operation of data stored in the register, the results would be the same as the results obtained by another execution such as arithmetic shift left. Such a process is also hardly required, so that a single operation code can be made to be assigned two different processes, like the above example.

[0014] Thus, according to the above constructions, various types of operations can be executed in a short instruction word length, and the sizes of programs can be minimised.

BRIEF DESCRIPTION OF THE DRAWINGS

[0015] These and other objects, advantages and features of the invention will become apparent from the following description thereof taken in conjunction with the accompanying drawings which illustrate a specific embodiment of the invention. In the drawings:-

FIG. 1 is a block diagram showing the construction of the main part of a data processing apparatus of this invention.

FIG. 2 is an illustration showing the detailed construction of a register array.

FIG. 3 is an illustration showing the detailed construction of a status register.

FIG. 4 shows examples of the constructions of instruction codes.

FIG. 5 is a table showing examples of assigning different operations to different instruction codes.

DESCRIPTION OF THE PREFERRED EMBODIMENT

[0016] The embodiment of this invention is described as follows with reference to FIGS. 1 through 5.

[0017] FIG. 1 includes components 11-17. The instruction buffer 11 temporarily stores instructions fetched from an unillustrated memory. The instruction register 12 holds instructions outputted from the instruc-

tion buffer 11 until they are decoded or executed. The register designation code identification inquiry unit 13 inquires, when two register designation codes are included in one instruction, as to whether they are equal or not. The instruction decoder 14 decodes instructions held in the instruction register 12 and outputs a control signal to control the operation of each unit of the data processing apparatus. The instruction decoder 14, when the operation code included in an instruction is a predetermined one, conducts different controls (described later) depending on the results inquired by the unit 13: they are equal or different from each other. The detailed description of the construction of the circuit to conduct such control is omitted because the circuit can be a microprogram circuit or a hardware logic circuit as well-known processors can.

[0018] The register array 15, as shown in FIG. 2, has a data register group consisting of data registers D0-D3 each having 16-bit length, an address register group consisting of address registers A0-A3, a program counter PC holding the storing address of the instruction in execution, and a status register PSW (processor status word) indicating the status of the data processing apparatus.

[0019] The data registers D0-D3 are mainly used for storing data, as working storage. On the other hand, the address registers A0, A1, and A2 are used for storing addresses indicating a source operand, a destination operand, or a frame pointer, A3 being used for storing an address indicating a stack pointer. Although the address registers A0-A2 are usually assigned in the above order when a program is made, their hardware is constructed so that the assignment of the address registers A0-A2 can be flexible and the assignment of the address register A3 can be fixed to use for a stack pointer.

[0020] All the data registers D0-D3 and all the address registers A0-A3 are designated by register designation codes of 00-11 (binary characters) in the instruction codes.

[0021] The status register PSW, as shown in FIG. 3, consists of a trace flag T designating a single step execution of a program during an operation in a debug mode, interruption mask flags IM3-IM0 designating 16-level interruption masks, an overflow flag V, a carry flag C, a negative flag N, and a zero flag Z all indicating the results of operations.

[0022] The execution unit 16 inputs data and addresses outputted from the register array 15, immediate data outputted from the instruction buffer 11, data fetched from the unillustrated memory through the internal data bus 17 and executes predetermined operations such as arithmetic operations, logical operations, and moves according to a control signal from the instruction decoder 14. The results obtained by the execution of the execution unit 16 are moved to the register array 15 and the memory through the bus 17 and held by specific registers corresponding to the instructions or by other

units.

[0023] The data processing apparatus of this embodiment is further provided with a data bus interface connected to the data bus of the unillustrated external portion of the data processing apparatus, an address bus interface connected to an address bus, and other units. These units are not described here because they are not directly related to this invention.

[0024] The following is a description of the construction of the instruction code assigned to the data processing apparatus and an example of assigning operations to operation codes with reference to FIGS 4 and 5. The instruction system described here is a read/store instruction system, which can simplify the construction of the apparatus and easily increase the processing speed, that is, an instruction system capable of mainly moving data between each register and the memory.

[0025] FIG. 4 shows examples of the constructions of instruction codes.

[0026] The code (a) consists of a 4-bit operation code, two 2-bit register designation codes, Dn and Dm, or Dn and Am (both n and m are integers of 0-3).

[0027] The code (b) includes either 8-bit immediate data (imm8), 8-bit displacement of an address (d8), or an 8-bit absolute address (abs8), in addition to the same operation code and the register designation codes as the code (a).

[0028] The code (c) consists of a 6-bit operation code and a 2-bit register designation code (Dn).

[0029] The code (d) consists of an 8-bit operation code and an appended portion of either 8, 16, or 24 bits such as an expansion instruction, immediate data, or addressing displacement.

[0030] In FIG. 5 showing an example of assigning instruction codes, the symbols "****" and "***" indicate that any one of the operations shown in the column of operation is executed according to the value of 3-bit or 2-bit respectively.

[0031] The following is a description of each instruction code according to each instruction type.

[0032] An instruction type [R-R] is a dyadic operation instruction between two data registers and is expressed by an instruction code "0***DnDm" (n=m). This instruction code indicates 8 different types of operation instructions as follows, depending on the 3-bit values, "****".

- (1) Moving data from a data register indicated by the data register designation code Dn (hereinafter referred to as data register Dn) to the data register Dm.
- (2) Adding data held in the above data registers without carries.
- (3) Subtracting the same with borrows.
- (4) Comparing the same.
- (5) Adding the same with carries.
- (6) Subtracting the same with borrows.

(7) Logically producing the same.

(8) Logically adding the same.

[0033] The results of each execution are stored in the data register Dm.

[0034] An instruction type [imm8-R] can be distinguished from the instruction type [R-R] because its two register designation codes are equal (n=m). The data register Dn data and 8-bit immediate data (imm8), which follows an instruction code, "0***DnDm" are processed by the same dyadic operation as the instruction type [R-R]. Even if 8-bit immediate data is used for operations, 16-bit data is inputted to the execution unit 16; however, its high-order 8 bits are provided with zero filling (for example in logical operations), sign bit filling (for example in arithmetic operations), or the like, according to the type of the operation.

[0035] The instruction types [LD] and [ST] are respectively a data move instruction (load) from the memory to the data register Dn and a reverse data move instruction (store). The sources and destinations in the memory are indicated by 4 types of addressing modes.

[0036] The symbol "@" in the column of operation in FIG. 5 is put before the value which is the address of the area to/from which data is moved, and the symbol "(,)" indicates the sum of the values of both sides of comma.

[0037] The addressing mode of each move instruction indicates the followings.

(1) When the source or destination of data is "@Am", it represents address register indirect addressing, which has a value stored in the address register Am as its address.

(2) When the source or destination of data is "@(Am, d8)", it represents address register indirect addressing with 8-bit displacement, which uses 8-bit displacement d8 following the instruction codes "1001DnAm" or "1101DnAm", and which has the sum of d8 and a value stored in the address register Am as its address.

(3) When the source or destination of data is "@(Am, D0)", it represents index addressing, which uses the address register Am (m=3) and the data register D0 (fixed), and which has the sum of a value stored in the address register Am and a value stored in the data register D0.

(4) When the address or addressee of data is "@abs8", it represents 8-bit absolute addressing, which has 8-bit absolute address abs8 following the instruction code as its address.

[0038] In the index addressing of (3) above, the address registers A0-A2 are exclusively used, the address register A3 being excepted. This is because this type of addressing is usually used to move data such as strings or arrays, which are rarely moved to the

stack area of the memory.

[0039] In the 8-bit absolute addressing of (4) above, the register designation code m=3 representing the address register A3 excepted in the index addressing is used as a dummy and an instruction code, where the data register D0 of the source or the destination is definitely used is assigned thereto. When data move is performed in 8-bit absolute addressing, a predetermined value such as "00000000" can be outputted as the high-order 8 bits out of 16-bit address from the data processing apparatus.

[0040] An instruction type [R] is a monadic operation instruction assigned to a single data register, and is indicated by the instruction code "1011**Dn" or "1111**Dn". This code indicates 8 different types of operation instructions: arithmetic shift right (ASR) and left (ASL), rotation right (ROR) and left (ROL), logical reverse (NOT), increment/decrement (INC/DEC), or program interruption (PI), depending on the 2-bit values, "****".

[0041] There are two groups of three instruction codes whose high-order 4 bits are respectively "1010" and "1110", which are not assigned any operation in the above example. These instruction codes are used to move their addresses to the address registers A0-A3, to assign unillustrated other instructions, to expand instruction codes to more than 16 bits, and other purposes. Thus, instructions with few frequency of use and with little effect on program sizes can be set as expansion instruction codes. The detailed description of such an expansion is omitted because it is not directly related to this invention.

[0042] When an instruction code whose most significant bit is zero, like "01110001" is inputted to the data processing apparatus constructed as above, the register designation code identification inquiry unit 13 compares the least and second least significant bits "01" to the third and forth least significant bits "00", and then outputs a signal indicating that n of Dn and m of Dm are not identical, to the instruction decoder 14.

[0043] The unit 14, then, outputs a control signal to the data registers D0 and D1, and makes them output data held in them to the execution unit 16 through the internal data bus 17.

[0044] The unit 14 also outputs to the execution unit 16 a control signal by which a logical add operation is executed on the basis of the second, third, and forth most significant bits of the instruction code, "111". Responding to this, the execution unit 16 executes a logical add operation of data outputted from the data register D0 and D1. The results are stored in the data register D1 through the internal data bus 17.

[0045] On the other hand, when the data processing apparatus is inputted a code "01110000" and the succeeding 8-bit immediate data, the register designation code identification inquiry unit 13 confirms n and m being equal.

[0046] Then, data held in the data register D0 and

8-bit immediate data accumulated in the instruction buffer 11 are inputted to the execution unit 16 through the internal data bus 17, and a logical add operation is executed and the results are stored in the data register D0.

[0047] In other words, according to the conventional data processing apparatuses, if two identical register designation codes are used to execute the same operation as non-identical register designation codes, the value of, for example, the data register D0 will not change.

[0048] In contrast, according to the data processing apparatus of this invention, 16 types of instructions can be assigned in 3-bit field indicated by the symbol "****" in FIG. 5, by being made to execute a different operation in such a case. Consequently, various types of operations can be performed even if the basic instruction word length is 8 bits. However, some instructions may be designed to execute the same operations as non-identical register designation codes by, for example, using the execution of subtraction in the identical register designation codes as a clear instruction of the register.

[0049] In the above case, one register can be designated in a 2-bit field by dividing 8 registers in the register array 15 into two groups having different functions: the data registers D0-D3 and the address registers A0-A3.

[0050] Therefore, even if the 8-bit instruction code is made to include two register designation codes: a source register and a destination register, the type of the operation can be designated by the remaining 4 bits long; consequently, various types of operations can be executed in much shorter basic instruction word length. As a result, frequently used eight basic operations such as operations among data registers, operations between the data registers and immediate data, moves between the data registers and the memory, and single-operand operations assigned to one data register can be executed by an instruction having a short word length.

[0051] According to the registers and operation instructions of the above construction, the environment of minimum requirements can be provided in order to effectively execute processing by a high-level language such as C language, the sizes of programs can be minimized, and the speed of process can be increased.

[0052] Although the immediate data and the absolute address in the instruction codes are 8 bits in length and are expanded to 16 bits long according to the instruction in the above embodiment, 16-bit immediate data may be designed to be used if necessary.

[0053] Although the instruction code is decoded into an instruction including an appended portion as immediate data when the two register designation codes are identical in the above embodiment, it may be designed to be decoded into an instruction including an appended portion as an immediate address or an instruction including no appended portion.

[0054] Furthermore, this invention is applicable also to the constructions of registers and instruction codes, the types of operations, and bit assignments of operation codes, all of which are different from those in the above embodiment in order to adjust to the various use of the data processing apparatuses.

[0055] Moreover, the instruction decoder 14 may be constructed so that it decodes the two register designation codes as different instructions depending on whether they are identical or not, instead of providing the register designation code identification inquiry unit 13.

[0056] Although the present invention has been fully described by way of examples with reference to the accompanying drawings, it is to be noted that various changes and modifications will be apparent to those skilled in the art.

Claims

1. A data processing apparatus comprising:

decode means (14) for decoding an instruction code including an operation code and a first and a second register designation code; and instruction execution means (13,16) for executing an appropriate operation according to the results of the decoding by the decode means (14), wherein the instruction execution means (13,16) executes a first operation based on the operation code when the second register designation code is different from the first register designation code, and executes a second operation based on the operation code when the second register designation code is equal to the first register designation code, characterised in that the first operation is executed with first and second registers (15) designated by the first and second register designation codes, the second operation is executed with the first register (15) designated by the first register designation code and immediate data included in the instruction code, and the first and second operations correspond to the same operation with different addressing modes.

2. The data processing apparatus of claim 1, wherein the instruction execution means (13,16) comprises:

inquiry means (13) for inquiring as to whether the second register designation code is different from or equal to the first register designation code; and control means (16) for controlling the first and second operations so that they can be selective according to the results inquired by the inquiry means (13).

3. The data processing apparatus of claim 1, further comprising:

m data registers (15) to hold data; and n address registers (15) to hold addresses; wherein the first and second register designation codes designate registers (15) to be processed; the operation code indicates data move operations among the data registers (15) and a memory connected to the apparatus, and operations between the data registers(15); and the first and second register designation codes have $\log_2 m$ and $\log_2 n$ lengths respectively, and the instruction code has a predetermined length which is the sum of the lengths of the operation code and the first and second register designation codes.

20 4. The data processing apparatus of claim 3, wherein the predetermined length of the instruction code is 8 bits.

25 5. The data processing apparatus of claim 3 or 4, wherein the instruction code further includes an appended code when the first and second register designation codes are equal.

30 6. The data processing apparatus of claim 5, wherein the appended code indicates the immediate data.

Patentansprüche

1. Datenverarbeitungsvorrichtung mit:

einer Decodiereinrichtung (14) zum Decodieren eines BefehlsCodes einschließlich von einem Operationscode und einem ersten und zweiten Registerangabecode; und

einer Befehlausführungseinrichtung (13, 16) zum Ausführen einer entsprechenden Operation in Übereinstimmung mit den Ergebnissen der Decodierung durch die Decodiereinrichtung (14), wobei die Befehlausführungseinrichtung (13, 16) eine erste Operation auf der Basis des Operationscodes ausführt, wenn sich der zweite Registerangabecode von dem ersten Registerangabecode unterscheidet, und eine zweite Operation auf der Basis des Operationscodes ausführt, wenn der zweite Registerangabecode gleich dem ersten Registerangabecode ist, dadurch gekennzeichnet, daß die erste Operation mit dem durch den ersten und den zweiten Registerangabecode angegebenen ersten und zweiten Register (15) ausgeführt wird, während die zweite Operation mit dem durch den ersten

- Registerangabecode angegebenen ersten Register (15) und den im Befehlscode enthaltenen Direktdaten ausgeführt wird, wobei die erste und die zweite Operation derselben Operation in jeweils einem anderen Adressierungsmodus entsprechen.
2. Datenverarbeitungsvorrichtung nach Anspruch 1, wobei die Befehlausführungseinrichtung (13, 16) umfaßt:
- eine Untersuchungseinrichtung (13), um zu untersuchen, ob der zweite Registerangabecode dem ersten Registerangabecode gleich ist oder sich von demselben unterscheidet, und eine Steuereinrichtung (16), um die erste und die zweite Operation zu steuern, so daß sie in Übereinstimmung mit den durch die Untersuchungseinrichtung (13) erhaltenen Ergebnissen ausgewählt werden können.
3. Datenverarbeitungsvorrichtung nach Anspruch 1, welche weiterhin umfaßt:
- m Datenregister (15) zum Speichern von Daten, und n Adreßregister (15) zum Speichern von Adressen,
- wobei der erste und der zweite Registerangabecode jeweils zu verarbeitende Register (15) angeben,
- der Operationscode Datenverschiebungsoperationen zwischen den Datenregistern (15) und einem mit der Vorrichtung verbundenen Speicher sowie Operationen zwischen den Datenregistern (15) angibt, und
- der erste und der zweite Registerangabecode jeweils \log_2 und $\log_2 \cdot n$ Längen aufweisen, wobei der Befehlscode eine vorbestimmte Länge aufweist, die der Summe aus den Längen des Operationscodes und des ersten und zweiten Registerangabecodes entspricht.
4. Datenverarbeitungsvorrichtung nach Anspruch 3, wobei die vorbestimmte Länge des BefehlsCodes 8 Bit beträgt.
5. Datenverarbeitungsvorrichtung nach Anspruch 3 oder 4, wobei der Befehlscode weiterhin einen angehängten Code umfaßt, wenn der erste und der zweite Registerangabecode gleich sind.
6. Datenverarbeitungsvorrichtung nach Anspruch 5,
- wobei der angehängte Code die Direktdaten angibt.
- Revendications**
5. 1. Appareil de traitement de données comprenant :
- des moyens de décodage (14) pour décoder un code d'instruction comprenant un code d'opération et un premier et un second codes de désignation de registre ; et des moyens d'exécution d'instruction (13, 16) pour exécuter une opération appropriée selon les résultats du décodage par les moyens de décodage (14), dans lequel les moyens d'exécution d'instruction (13, 16) exécutent une première opération basée sur le code d'opération lorsque le second code de désignation de registre est différent du premier code de désignation de registre, et exécutent une seconde opération basée sur le code d'opération lorsque le second code de désignation de registre est égal au premier code de désignation de registre ; caractérisé en ce que la première opération est exécutée avec des premier et second registres (15) désignés par les premier et second codes de désignation de registre, la seconde opération est exécutée avec le premier registre (15) désigné par le premier code de désignation de registre et les données immédiates incluses dans le code d'instruction, et les première et seconde opérations correspondent à la même opération avec des modes d'adressage différents.
2. Appareil de traitement de données selon la revendication 1, dans lequel les moyens d'exécution d'instruction (13, 16) comprennent :
- des moyens d'interrogation (13) pour interroger si le second code de désignation de registre est différent ou égal au premier code de désignation de registre ; et des moyens de commande (16) pour commander les première et seconde opérations de telle sorte qu'elles peuvent être sélectives en fonction des résultats obtenus par les moyens d'interrogation (13).
3. Appareil de traitement de données selon la revendication 1, comprenant en outre :
- m données de registre (15) pour retenir des données ; et n registres d'adresse (15) pour retenir des adresses ; dans lequel les premier et second codes de désignation de registre désignent des registres

(15) à traiter ;

le code d'opération indique les opérations de déplacement de données parmi les registres de données (15) et une mémoire connectée à l'appareil, et des opérations entre les registres de données (15) ; et

les premier et second codes de désignation de registre ont respectivement des longueurs de $\log_2 m$ et $\log_2 n$, et le code d'instruction a une valeur prédéterminée qui est la somme des longueurs du code d'opération et des premier et second codes de désignation de registre.

5

10

4. Appareil de traitement de données selon la revendication 3, dans lequel la longueur prédéterminée du code d'instructions est 8 bits.

15

5. Appareil de traitement de données selon la revendication 3 ou 4, dans lequel le code d'instructions comprend en outre un code annexé lorsque les premiers et seconds codes de désignation de registre sont égaux.

20

6. Appareil de traitement de données selon la revendication 5, dans lequel le code annexé indique les données immédiates.

25

30

35

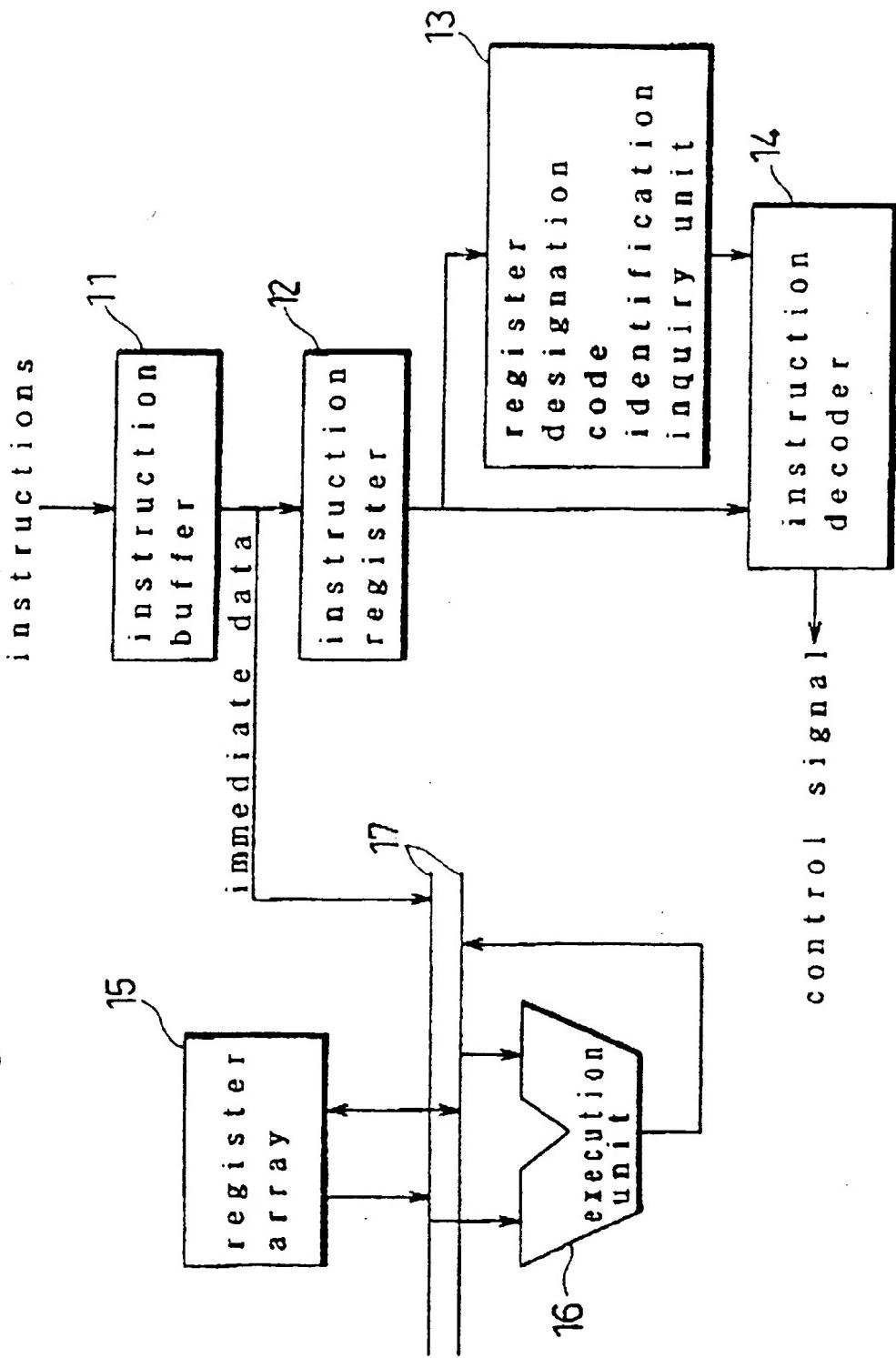
40

45

50

55

Fig. 1



THIS PAGE BLANK (uspto)

Fig. 2

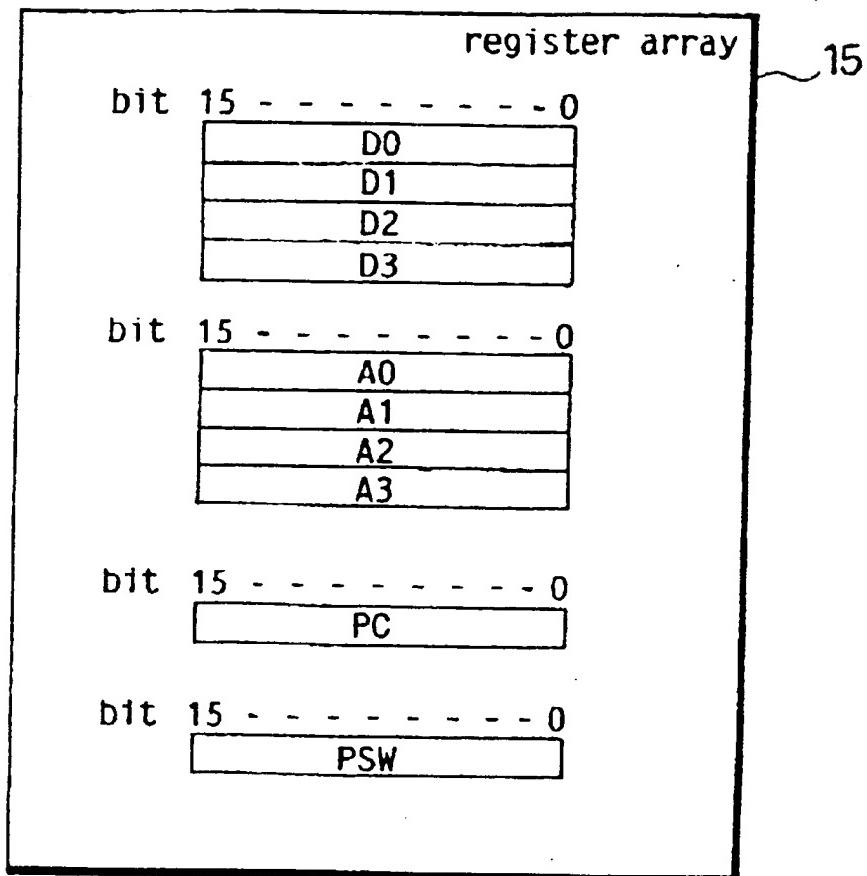
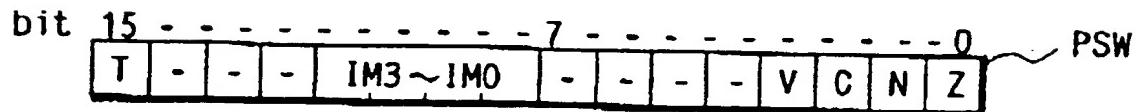


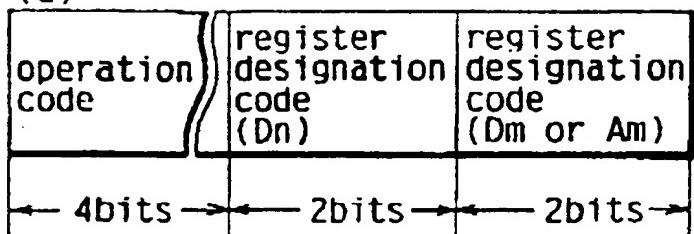
Fig. 3



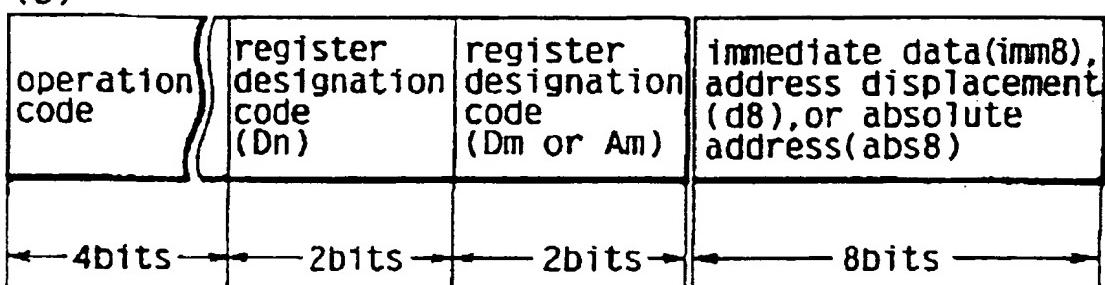
THIS PAGE BLANK (USPTO)

Fig. 4

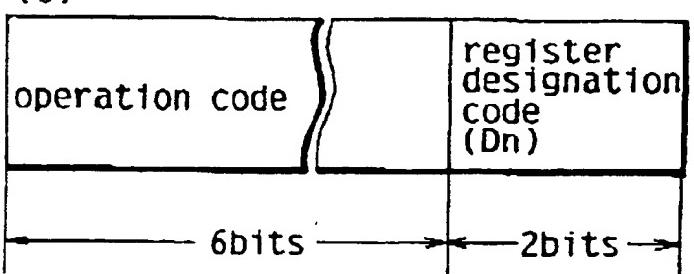
(a)



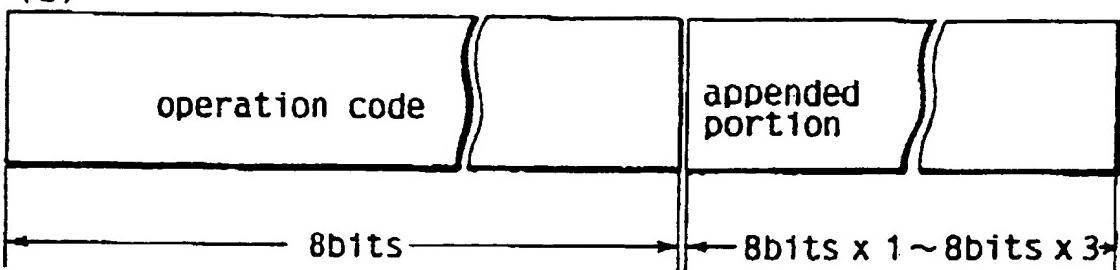
(b)



(c)



(d)



THIS PAGE BLANK (usPro)

Fig. 5

Instruction code	Constructions of instruction codes	Types of instructions	Operations
0 *** Dn Dm (n ≠ m)	(a)	[R - R]	according to the values of *** MOV ADD SUB CMP ADDC SUBC AND OR
0 *** Dn Dm + imm8 (n = m)	(b)	[imm8 - R]	same as above
1000 Dn Am	(a)	[LD] 	Addressees ③ Am Move from memory to data registers ③ (Am, d 8) ③ (Am, D 0) ③ abs 8
1001 Dn Am + d 8	(b)		
1010 Dn Am (m ≠ 3)	(a)		
1010 Dn Am + abs 8 (n = 0, m = 3)	(b)		
1100 Dn Am	(a)	[ST] 	Addressees ③ Am Move from data registers to memory ③ (Am, d 8) ③ (Am, D 0) ③ abs 8
1101 Dn Am + d 8	(b)		
1110 Dn Am (m ≠ 3)	(a)		
1110 Dn Am + abs 8 (n = 0, m = 3)	(b)		
1011 ** Dn	(c)	[R]	according to the values of **, ASR ASL ROR ROL
1111 ** Dn	(c)		according to the values of **, NOT INC DEC PI

THIS PAGE BLANK (USPTO)